An Ecological Forecasting Agent

Keith Golden,
Ramakrishna Nemani,
Wanlin Pang,
Petr Votava,
NASA Ames
Oren Etzioni,
U. Washington

3/28/2005

Project goals

- Make data analysis faster and cheaper
- Increase use of NASA data by removing barriers to data access
  - Cope with data heterogeneity
- Support code reuse and rapid application development
- Support multiple applications, users
  - Including fire and health domains
- Improve QOS
  - Always provide an answer
  - Tell user how good it is, where it came from

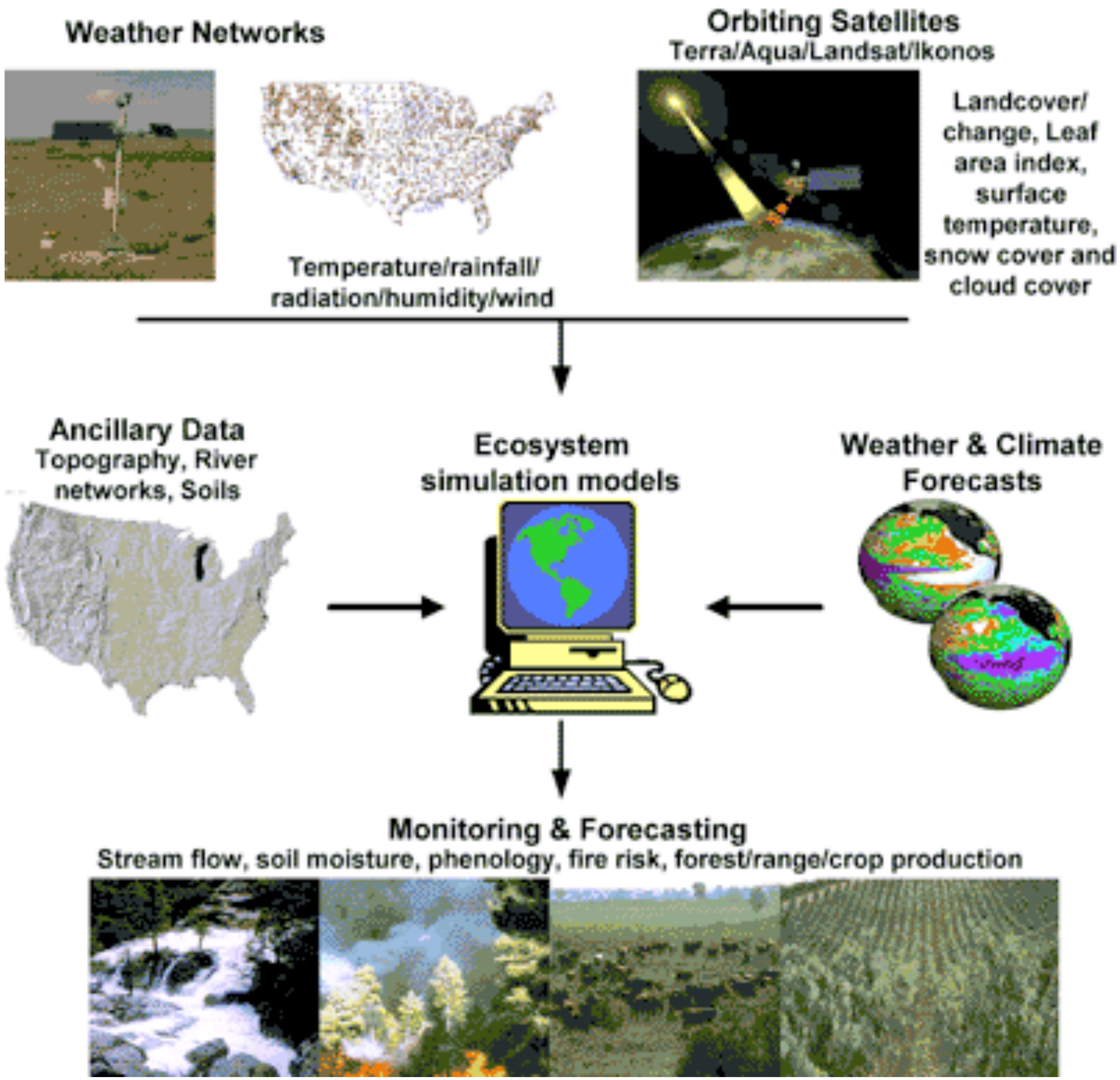
Team members

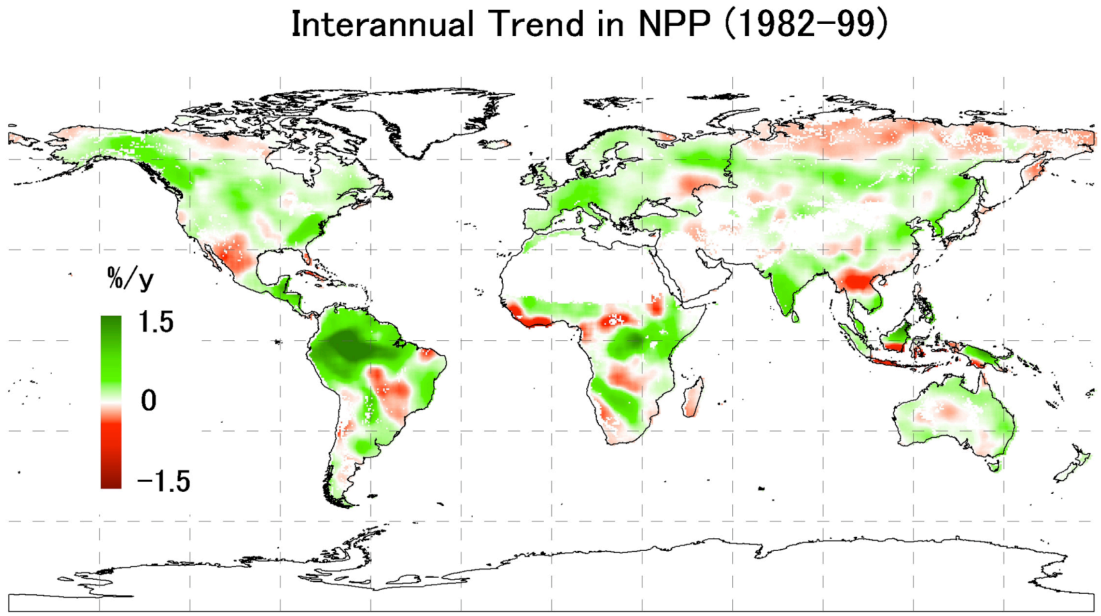
- Keith Golden -- PI, Planner
- Wanlin Pang -- Constraint Reasoning
- Rama Nemani -- Science
- Petr Votava -- TOPS implementation
- Oren Etzioni -- Natural Language Interface
- David Danks -- Automated discovery of models
- Forrest Melton -- Project management
- Andy Michaelis -- TOPS

Agents

- An agent is an intelligent assistant
  - E.g., travel agent
- Provides *goal-oriented* interface
  - You say **what** to do, not **how** to do it.
  - Agent has the knowledge to figure out **how**.
- Copes with uncertainty and error robustly.
  - Obtains information needed to complete task
  - Tries something else when encountering failure

Terrestrial Observation and Prediction System





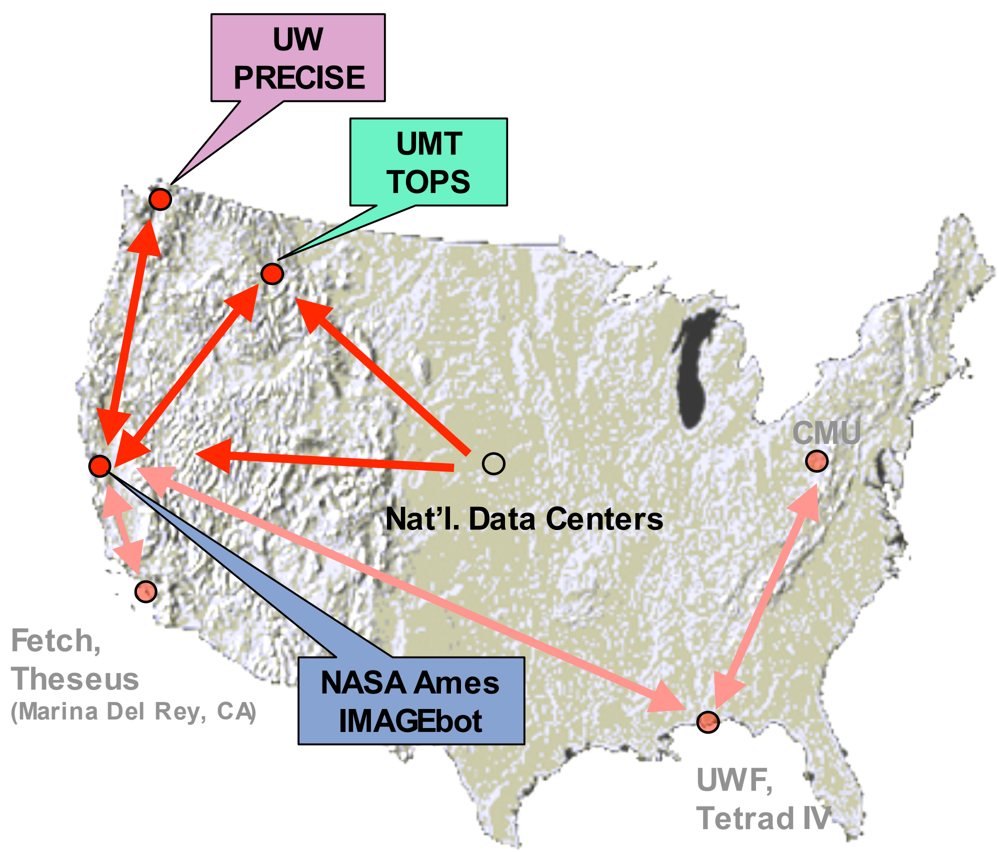
- Climate-driven increases in global terrestrial net primary production from 1982 to 1999. *Science*, 300, 1650 (2003).
  - 3 research assistants for 12 months
  - processed <15 GB of data
  - data preparation >80%
- EOSDIS:
  - Generates ~3 tera-bytes of data a day.
  - Currently holds 2 peta-bytes
  - 1 day = 2 years of HUBBLE Space Telescope

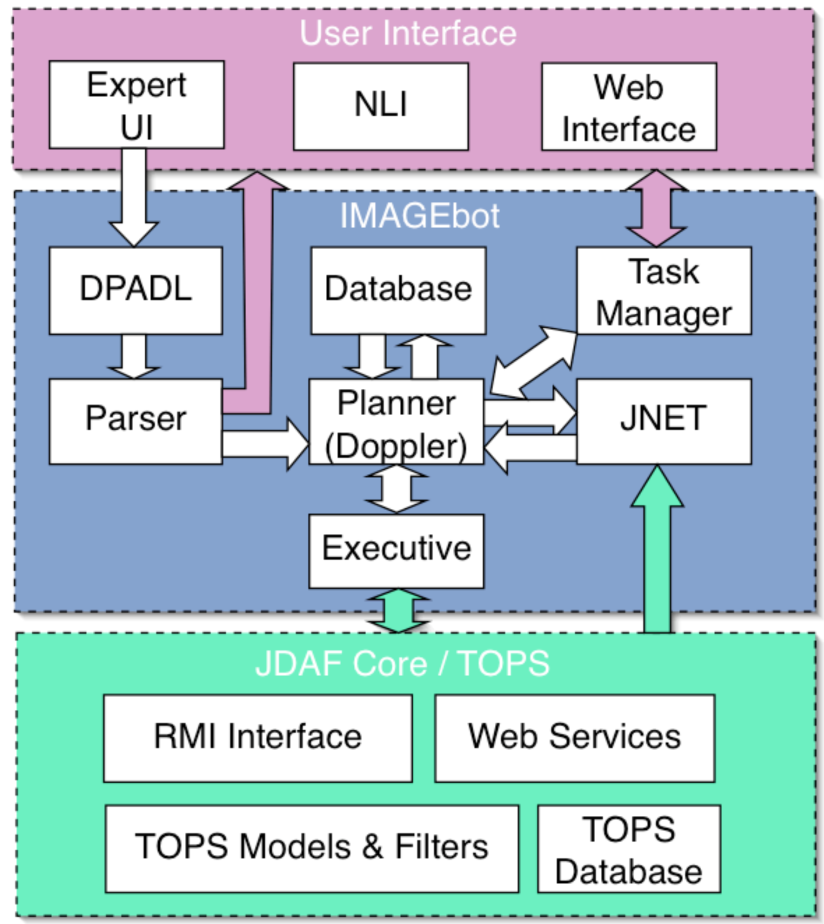
Dimensions of Autonomy

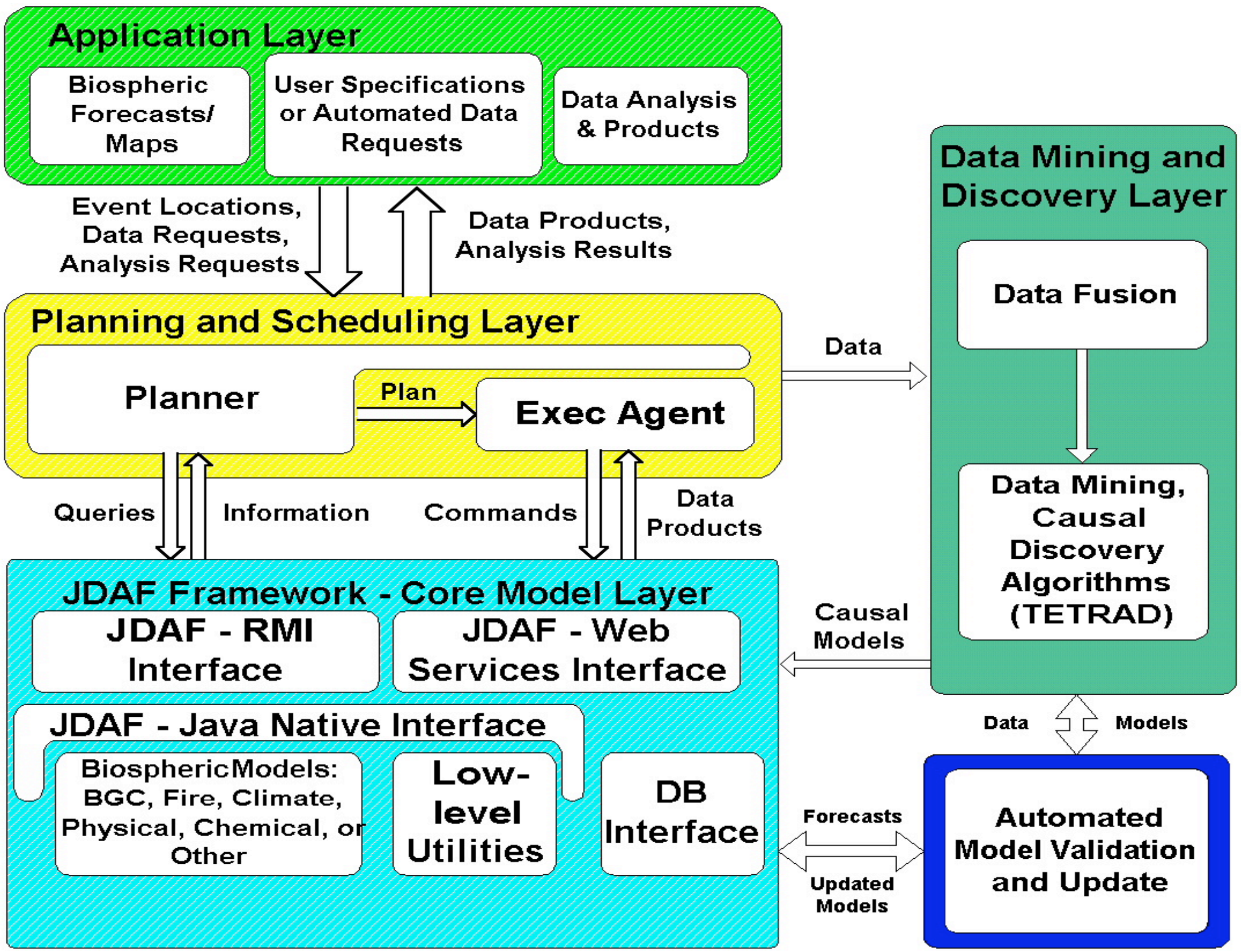
- Range of controllable behavior
  - Automate the generation of the forecasts, analysis of the results, and model adjustments
  - Generate and record “meta-data” to facilitate later searches
  - Integrate of new models and data sources in “plug & play” fashion
- Command specificity
  - High-level goals (descriptions of desired data products)
  - No premature commitment to specific format, resolution, projection, etc.
- Execution Robustness
  - Adapt to changes and recover form failures
  - Out of several sources of the same data use the “best” available one

Ames Research Center

Distributed Agent Architecture






Application Layer
Biospheric Forecasts/ Maps
User Specifications or Automated Data Requests
Data Analysis & Products
Event Locations, Data Requests, Analysis Requests
Data Products, Analysis Results
Planning and Scheduling Layer
Planner
Plan
Exec Agent
Queries
Information
Commands
Data Products
Data
Data Mining and Discovery Layer
Data Fusion
Data Mining, Causal Discovery Algorithms (TETRAD)
JDAF Framework - Core Model Layer
JDAF - RMI Interface
JDAF - Web Services Interface
JDAF - Java Native Interface
BiosphericModels: BGC, Fire, Climate, Physical, Chemical, or Other
Low-level Utilities
DB Interface
Causal Models
Data
Models
Forecasts
Updated Models
Automated Model Validation and Update

Planner capabilities

- Automate data collection and processing
- Integrate multiple data sources
  - Now: satellite, weather stations
  - Eventually: UAV, ground sensors
- Support flexibility and robustness
  - Constraints allow consistency without premature commitment to data source, file format, resolution, projection.
  - Preferences allow planner to produce best data possible
    - According to user's priorities
    - Subject to available data sources
  - Multiple options + backtracking means errors or dropouts won't result in failure.
- Ease application development through abstraction

Data Processing Action Description Language

- Declarative representation of data sources, models and data product specifications
  - No need to specify scripts to process data
  - Modular, plug & play design
- Actions describe data-processing operations
  - Earth system models, data filters, composition, etc.
  - Any number of inputs and outputs
- Arbitrary constraints over any static type
- Integration with Java
  - Embedded Java code
    - Integration with JDAF or other systems
    - Action execution
    - “Procedural” constraints
  - Parameters include Java objects

Ames Research Center

Desired “planning language” properties

- Easily useable by planners
  - Declarative semantics
  - States, actions
  - As feature-poor as possible
- Compatible objectives?
  - Commonality: state, state change
    - Variables
    - Actions ≈ Procedures/methods
    - Preconditions ≈ tests
    - Conditional effects ≈ conditional instructions
  - “Compile” to simpler language

DPADL: Data Processing Action Description Language

- Object oriented, C++/Java-like syntax
  - Inheritance
  - Primitive types and objects
  - Object creation, copying, modification
- Integration with Java
  - Embedded Java code
    - Action execution
    - “Procedural” constraints
  - Parameters include Java objects
- Actions describe data-processing operations
  - Any number of inputs and outputs
  - Causal, declarative representation of data filters
- Constraints over any static type

Types

- Can inherit from objects or primitive types

```
static type Filename instanceof String;
```

- Can be defined by list of members

```
static type ImageFormat =
    {"JPG", "GIF", "TIFF", "PNG", "XCF", … };
static type ProjectionType =
    {LAZEA=11, GOODE_HOMOL=24, ROBINSON=21, … };
```

- Can represent complex data structures

```
type Image instanceof Object {
    static int xSize;
    static int ySize;
    PixelValue pixelValue(int x, int y);
    …
}
```

Functions

- Can be static or fluent

```
fluent float temperature(float lon, float lat);
static float sin(float x);
```

- Are the atoms of the language
  - Attributes are functions of their objects
  - Infix operators are functions (C++ style overloading)

```
static String operator+(string s1, string s2);
```

  - Relations are boolean functions

```
static boolean operator<(float r1, float r2);
```

  - Global variables are fluent functions with no arguments

```
fluent Date currentDate;
```

- Can be targets of assignment

```
image1.pixelValue(x, y) := image2.pixelValue(y, x);
```

Ames Research Center

Constraints

Can be attached to types or functions

```
static type Filename isa string {
   constraint Matches(true, this, "~[/]+");
}
static string operator+ (string s1, string s2)
  {
    constraint Concat(value, s1, s2);
}
```

Can be specified using embedded Java code.

```
static type Tile isa object mapsto
  tops.modis.Tile {
   Instrument product {
    constraint {
      result(this) := $ this.getProduct() $;
    }
…
```

Constraints

```
static type Tile isa object mapsto tops.modis.Tile { …
  //true if this tile covers the specified location
  boolean covers(float lon, float lat) {
    constraint {
      {this}([lon], [lat], d=day, y=year,
             p=product, value)
        := {$ if(value)
               return tm.getTiles(lon.max, lat.min,
                                  lon.min, lat.max,
                                  d, y, p);
               else return null; $ };
    }
  …
```

Actions

```
action threshold (unsigned thresh) {
  input BwImage in;
  output BwImage out copyof in;
  forall unsigned x, unsigned y;
  effect when (x < in.xSize && y < in.ySize) {
              when (in.valueAt(x, y) <= thresh) {
                 out.valueAt(x, y) = BLACK;
              } else {
                 out.valueAt(x, y) := WHITE;
              }
           }
  exec $ out = gfx.threshold(in, thresh); $;
}
```

Desired “programming language” properties

- Naturally describe domain concepts
  - Data semantics and syntax
  - Structured data files, complex data types
  - Object creation/copying
- Specify interfaces external environment
  - Execute plans
  - Obtain information (sensing)
- Easily usable by programmers
  - Similarity to known programming languages
  - As feature-rich as necessary

Ames Research Center

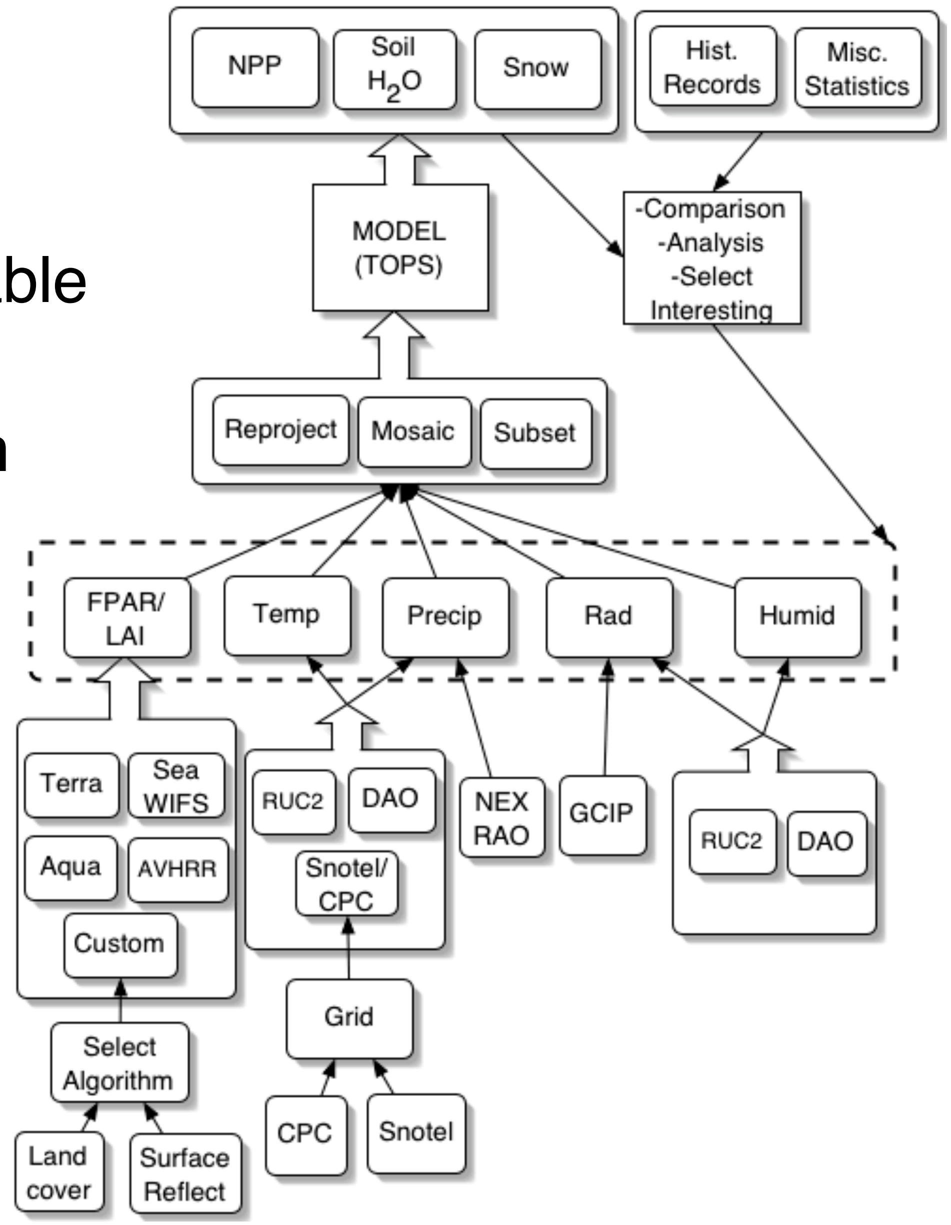
Data Goals and Metadata

- Data product specification
  - **What** information is contained
  - **How** information is encoded in data
  - **Where** the data files are stored/delivered
  - **When** the information pertains to
- Examples
  - I want an MPEG movie of yesterday's weather over the SF bay placed on our website
  - File dd010101.tar.gz is a compressed archive of the downlink directory as of Jan 1, 2001

What
How
Where
When

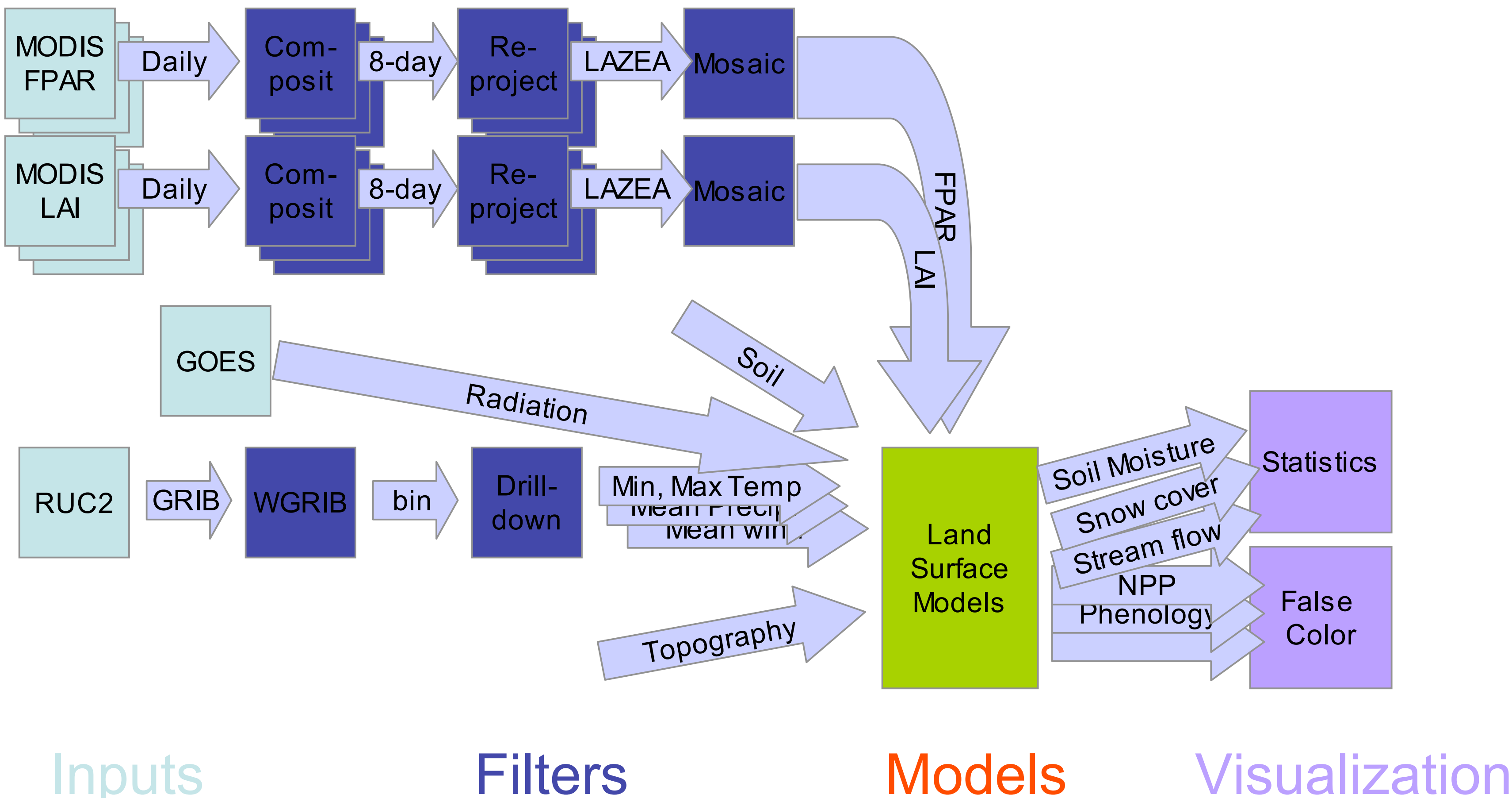
Planning for data processing

- Initial state = data available
- Goal = data to produce
- Plan = dataflow program
- Actions =
  - Earth system models
  - Data transformations
- Domain characteristics
  - Very large universes
  - Complex data structures
  - Lots of constraints
  - \+ Highly parallel



Ames Research Center

Dataflow plans

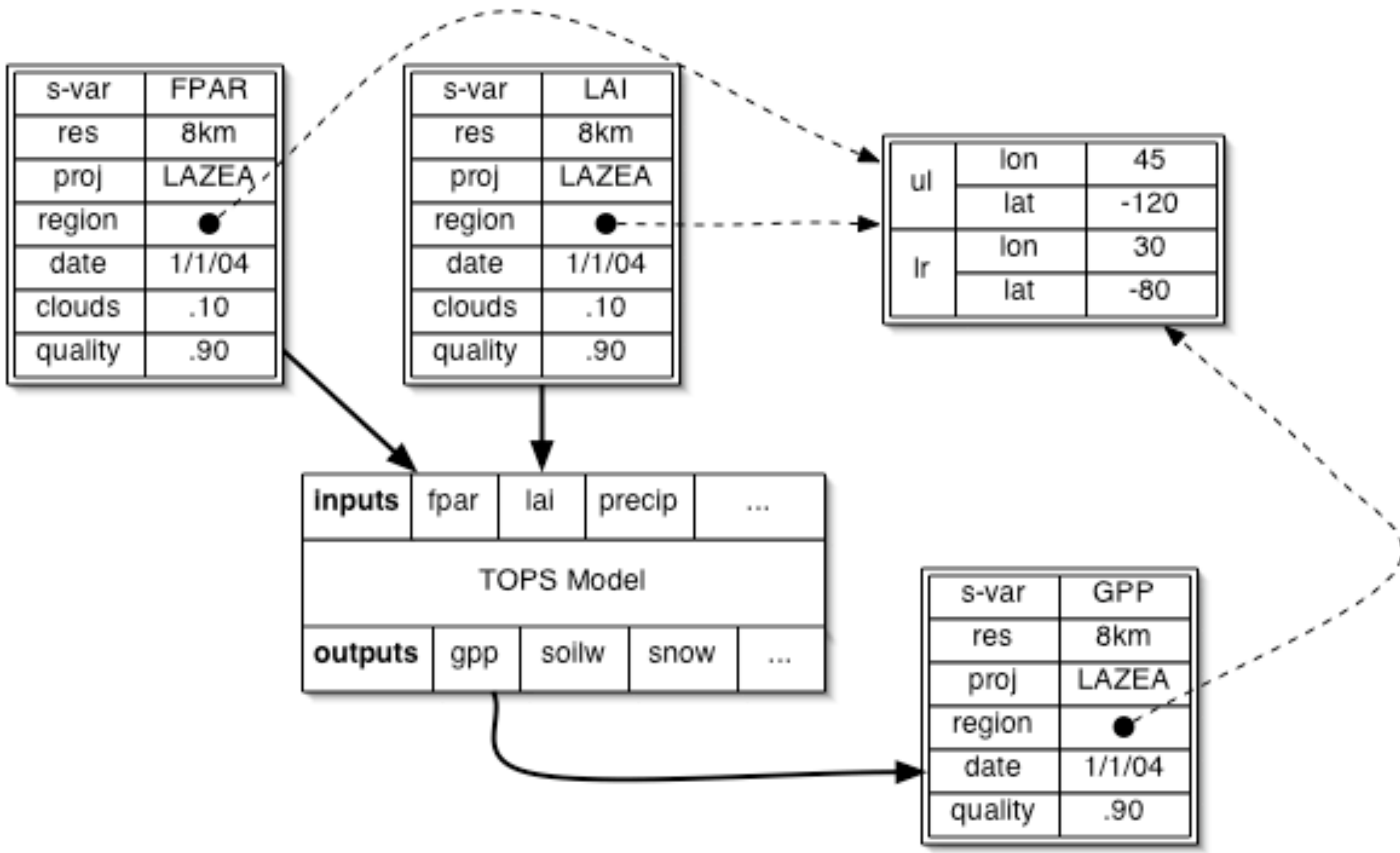


Ames Research Center

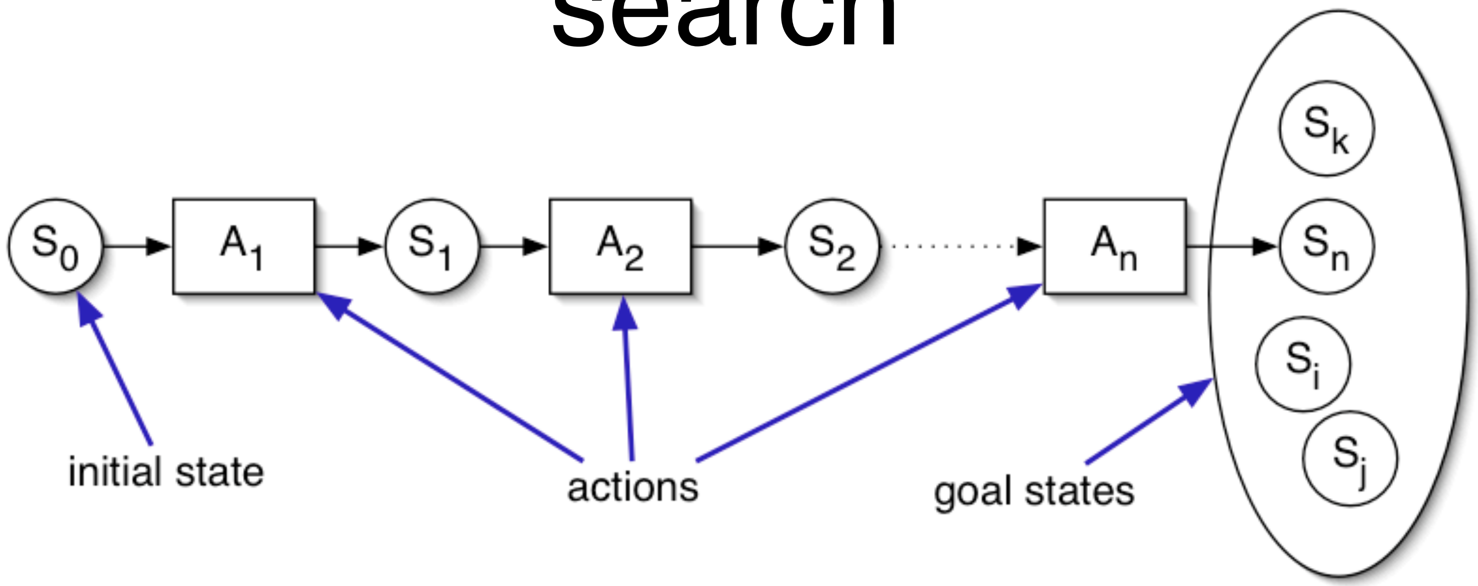
Input data choices

Terra-MODIS	FPAR/LAI	1 day	1km, 500m, 250m	global, since 2000
Aqua-MODIS	FPAR/LAI	1 day	1km, 500m, 250m	global, since 2002
AVHRR	FPAR/LAI	10 day	1 km	global, since 1981
SeaWIFS	FPAR/LAI	1 day	1xm x 4 km	global
DAO	temp, precip, rad, humid	1 day	1.25 deg x 1.0 deg	global, since 1980
RUC2	temp, precip, rad, humid	1 hour	40 km	USA
CPC	temp, precipication	1 day	point data	USA
Snotel	temp, precipitation	1 day	point data	USA
GCIP	radiation	1 day	0.5 deg	Continental
NEXRAD	precipitation	1 day	4 km	USA

Complex Data Structures

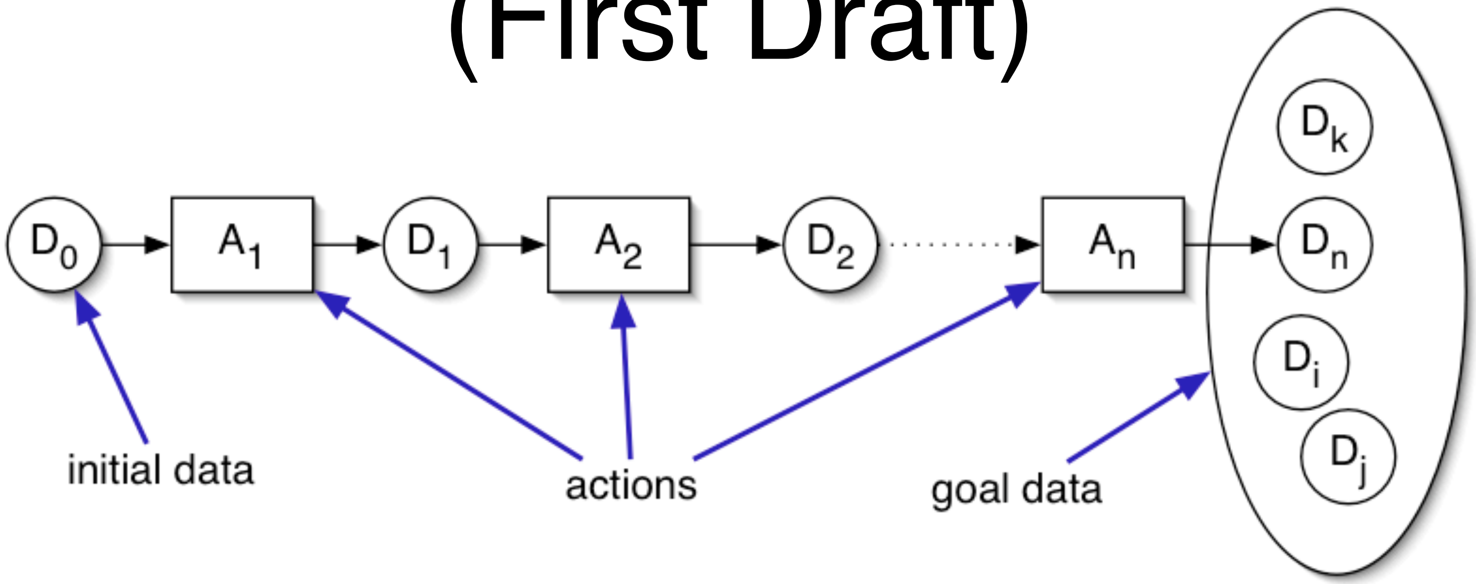


Planning as state-based search



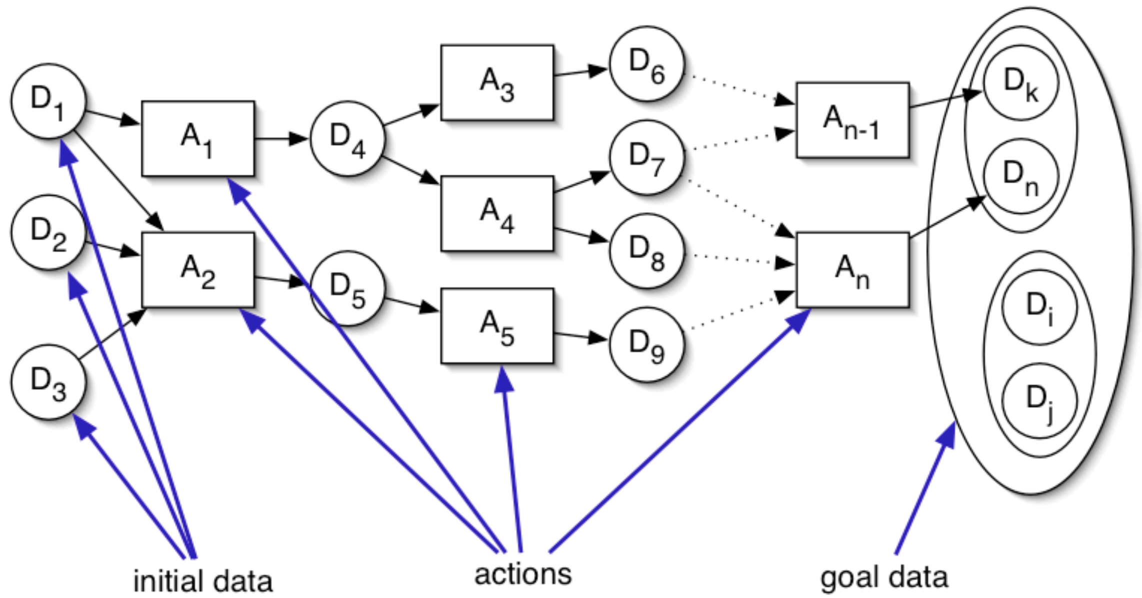
- Planning problem: $< s_0, \mathcal{A}, \mathcal{G}>$
  - s_0 = Initial state
  - $\mathcal{A}$ = Set of actions (state tranforms)
  - $\mathcal{G}$ = set of goal states
- Plan: sequence of actions from s_0 to state $s_n \in \mathcal{G}$

Planning as Data Production (First Draft)



- Planning problem: < D_0, $\mathcal{A}_{\mathcal{D}}$, $\mathcal{G}_{\mathcal{D}}$>
  - D_0 = Initial data
  - $\mathcal{A}_{\mathcal{D}}$ = Set of actions (data tranforms)
  - $\mathcal{G}_{\mathcal{D}}$ = set of goal data
- Plan: sequence of actions from D_0 to state $D_n \in \mathcal{G}_{\mathcal{D}}$

Planning as Data Production



- Planning problem: $<I_{\mathcal{D}}, \mathcal{A}_{\mathcal{D}}, \mathcal{G}_{\mathcal{D}}>$
  - $I_{\mathcal{D}}$ = Set of Initial data
  - $\mathcal{A}_{\mathcal{D}}$ = Set of actions (data tranforms)
  - $\mathcal{G}_{\mathcal{D}}$ = set of goal data sets
- Plan: DAG of actions from $I_{\mathcal{D}}$ to $\mathcal{D}_n \in \mathcal{G}_{\mathcal{D}}$

Planning approach

1. Perform graph analysis to derive
   - *distance heuristics*
   - *initial variable domains*
2. Convert planning problem into CSP
3. Search for solution using heuristics
   – Sensors represented as constraints
4. Execute plan, update database and replan if needed

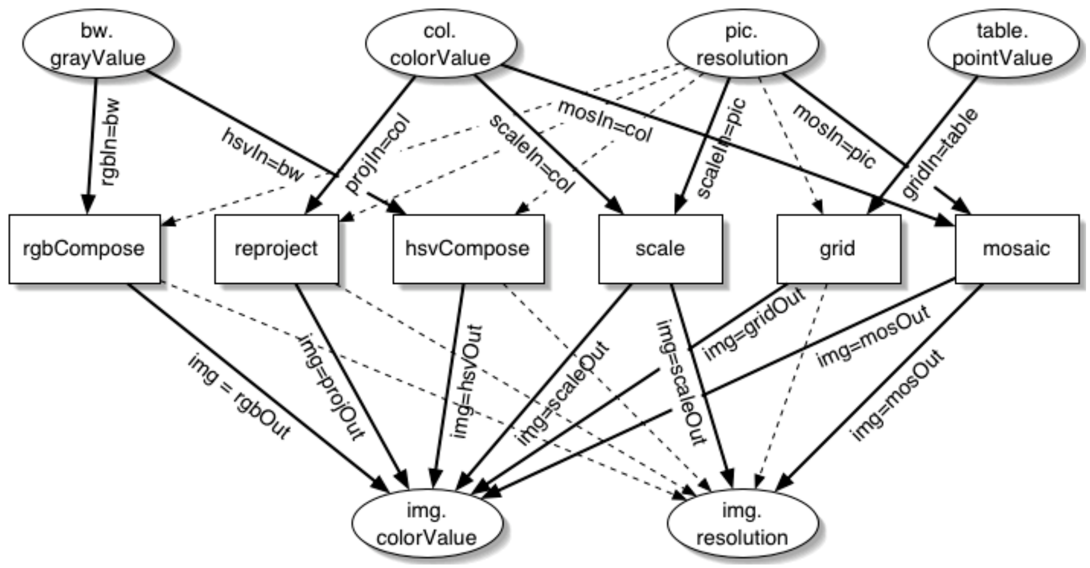
Preferences

- Not all “satisfactory” data products are equally good
- Constraints only allow us to specify what is (un)acceptable, not what is preferred
- Preferences allow users to specify what matters to them (resolution, quality, speed)

Optimality

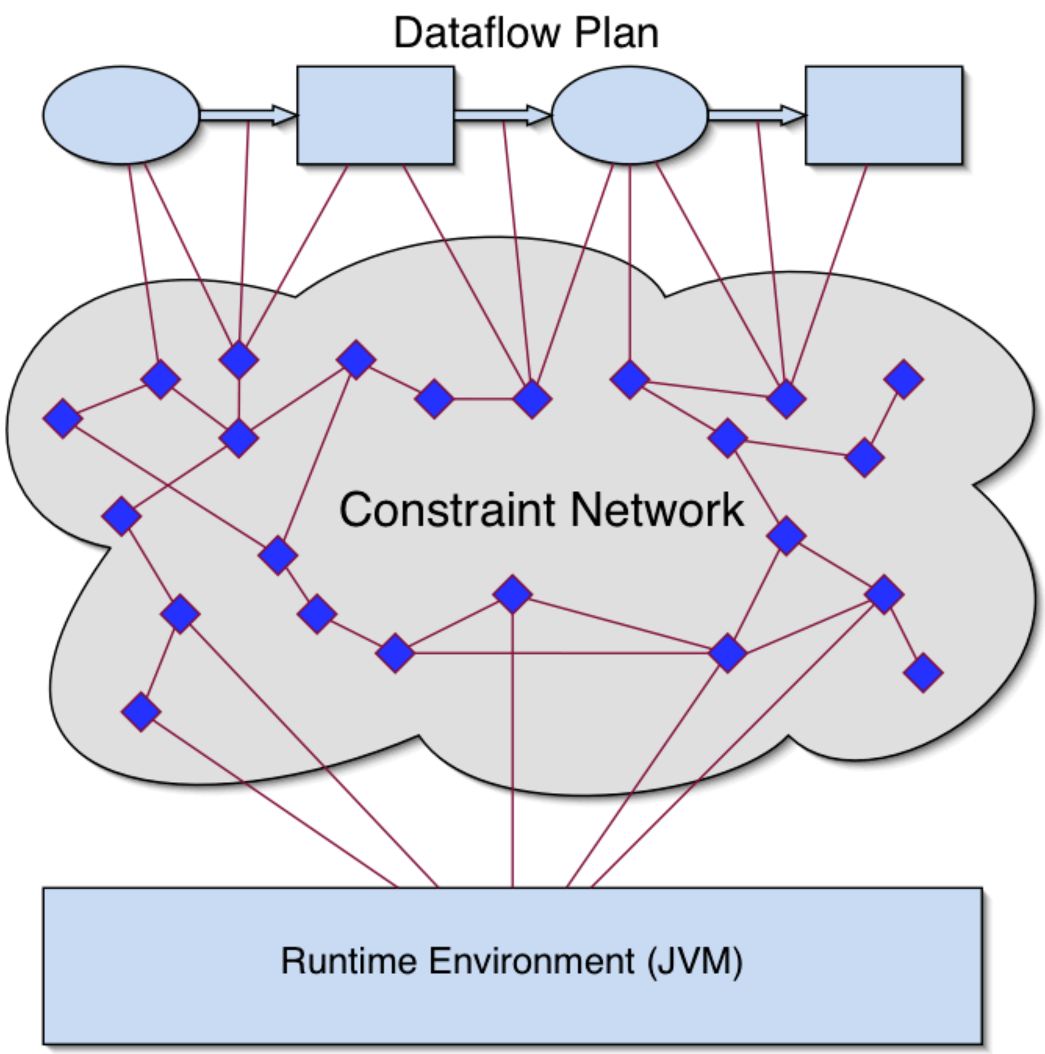
- IMAGEbot can produce the best possible data product given user preferences
  - Optimization handled as part of constraint search
  - Computationally hard, could take a while for some problems
- Pareto Optimality
  - When multiple unordered preferences are given
    - I want the best quality and I want it now
  - No preference could be satisfied more without satisfying another less
    - I can't give you better quality without spending more time
    - I can't give you a faster answer without sacrificing quality

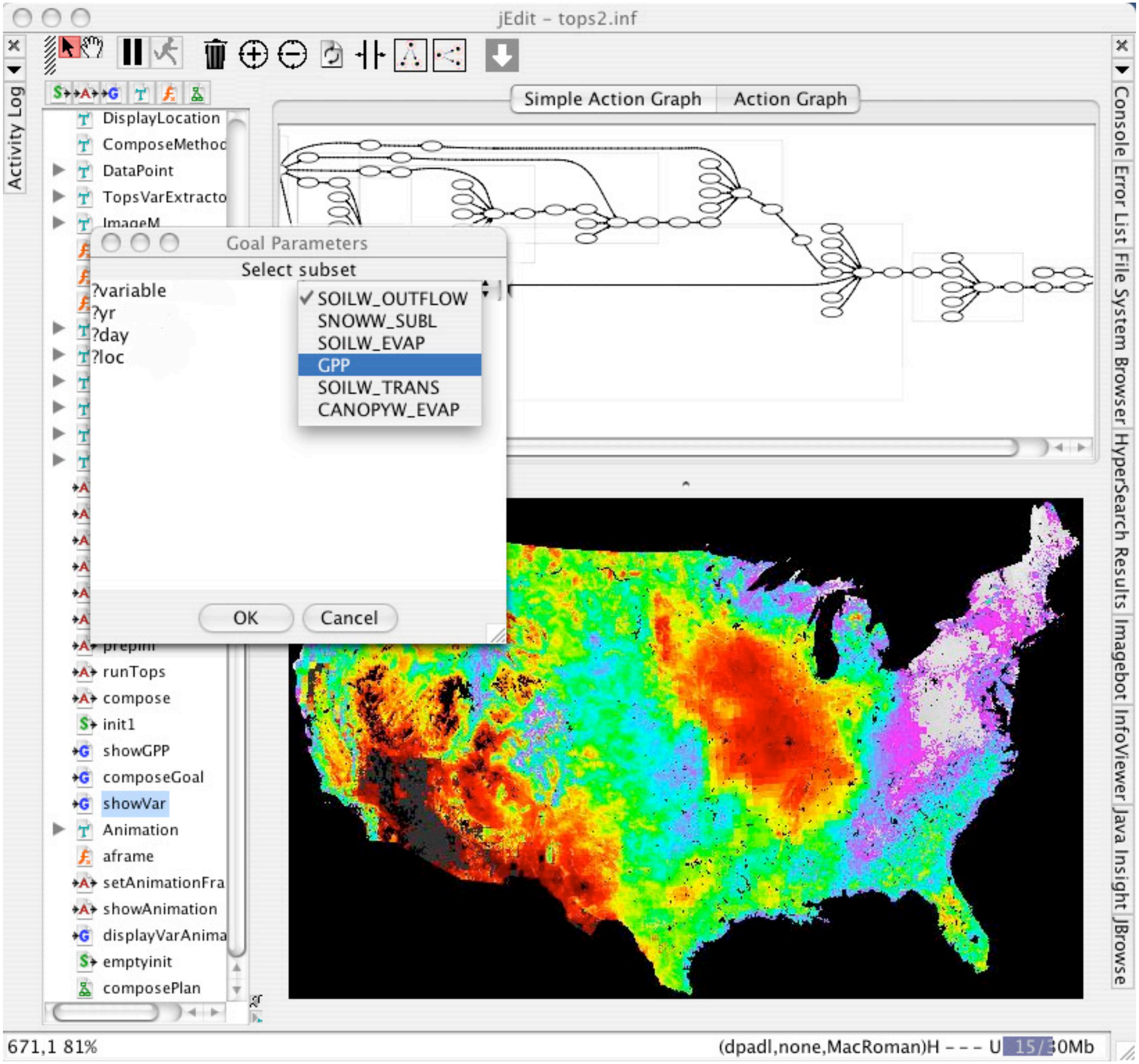
Problem graph analysis



- Obtain early information on
  - Possible parameter values
  - Appropriate data sources
  - Tradeoffs among optimization criteria

Constraints as glue



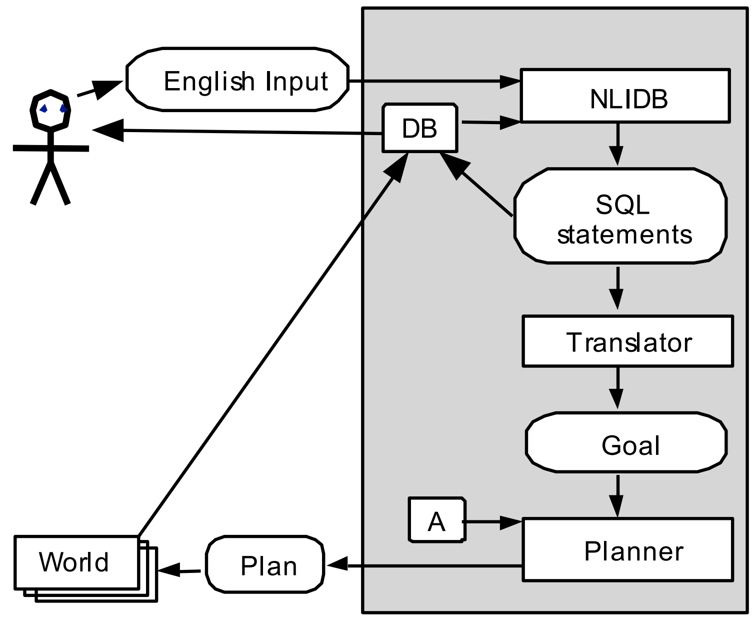

jEdit - tops2.inf
Activity Log
Simple Action Graph
Action Graph
Console Error List File System Browser HyperSearch Results Imagebot InfoViewer Java Insight JBrowse
DisplayLocation
ComposeMethoc
DataPoint
TopsVarExtracto
Goal Parameters
Select subset
?variable
?yr
?day
?loc
SOILW_OUTFLOW
SNOWW_SUBL
SOILW_EVAP
GPP
SOILW_TRANS
CANOPYW_EVAP
OK
Cancel
runTops
compose
init1
showGPP
composeGoal
showVar
Animation
aframe
setAnimationFra
showAnimation
displayVarAnima
emptyinit
composePlan
671,1 81%
(dpadl,none,MacRoman)H - - - U 15/30Mb

Ames Research Center

Natural Language Interface

Build on Precise NLIDB

1. English → SQL
2. Resolve ambiguity
3. SQL → DPADL
4. Resolve conflicts
5. Planner solves goal
6. User updated with result



PRECISE queries

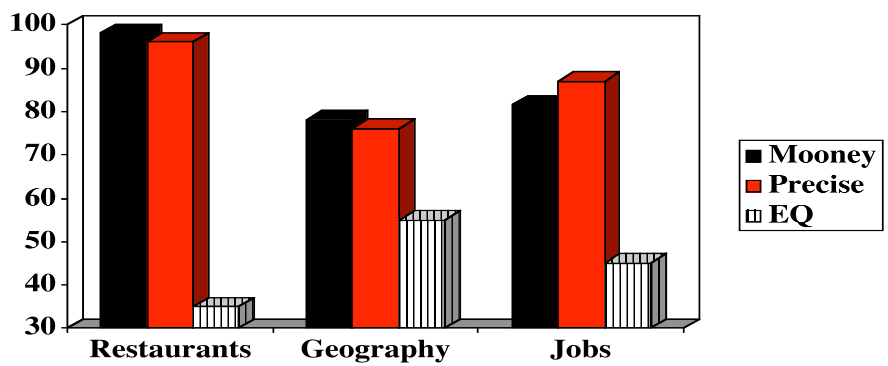
- What is the area of Alaska?
- What is the population of New York?
- Which are all the states that border Oregon?
- How many major cities are in Florida?
- How long is the Colorado River?
- What rivers traverse Indiana and Illinois?
- What rivers traverse Indiana or Illinois?
- What cities are in Texas and have a population of less than 100000 people?
- What is the largest city in the smallest state in the US?
- What states border the state with the largest population?

http://www.cs.washington.edu/research/projects/WebWare1/www/precise/precise.html

Comparison to state of art

Fraction Answered

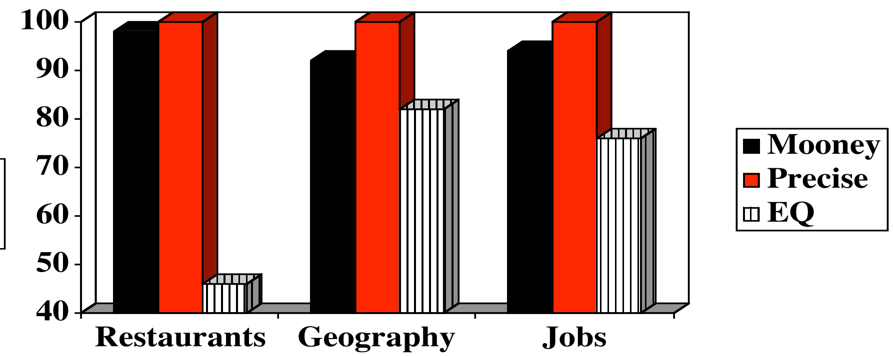
Recall = $Q_{answered}/Q$



Recall > 75 %

Error Rate

Precision = $Q_{correct}/Q_{answered}$



PRECISE made no mistakes on semantically tractable questions

Publications

- Popescu, A-M, A. Armanasu, D. Ko, A. Yates (2004). Modern Natural Language Interfaces to Databases: Composing Statistical Parsing with Semantic Tractability, *International Conference on Computational Linguistics*.
- Golden, K. and W. Pang (2003). Constraint reasoning over strings. *International Conference on Principles and Practice of Constraint Programming (CP 2003).*
- Golden, K., W. Pang, R. Nemani, P. Votava (2003). Automating the Processing of Earth Observation Data, *Proc. of the 2003 International Symposium on AI, Robotics and Automation for Space (i-SAIRAS)*.
- Golden, K. (2003). A Domain Description Language for Data Processing. International Conference on Automated Planning and Scheduling. Workshop on the Future of PDDL.
- Votava, P., R. Nemani, K. Golden, D. Cooke, H. Hernandez, C. Ma (2003). Parallel Distributed Application Framework for Earth Science Data Processing ScanGIS.
- P. Votava, K. Golden, R. Nemani. Planning for Earth Science Data Processing. In proceedings of International Symposium on Environmental Software Systems ISESS 2004.

Publications (cont'd)

- H. Hernandez, P. Votava, C. Ma. A Simple and Effective Parallelization of a Legacy Satellite- Processing Algorithm. Submitted to Journal of Scientific Computing. 2004.
- Golden, K., J. Frank (2002). Universal Quantification in a Constraint-Based Planner. *International Conference on AI Planning and Scheduling*.
- Nemani, R.R., C.D. Keeling, H. Hashimoto, W.M. Jolly, S.C. Piper, C.J. Tucker, R.B. Myneni and S.W. Running (2003). Climate-Driven Increases in Global Terrestrial Net Primary Production from 1982 to 1999. *Science*, June 6, 2003.
- Nemani, R.R., M.A. White, Lars Pierce, Petr Votava, Joseph Coughlan and S.W. Running (2003). Biospheric monitoring and ecological forecasting. *Earth Observation Magazine*, 12(2): 6-8.
- Golden, K. (2002). DPADL: An Action Language for Data Processing Domains. *International NASA Workshop on Planning and Scheduling for Space*.
- Votava, P., R. Nemani, C. Bowker, A. Michaelis, A. Neuschwander, J. Coughlan (2002). Distributed Application Framework for Earth Science Data Processing. In *Proceedings of IEEE International Geoscience and Remote Sensing Symposium* (IGARSS) 2002.

Related Work

- GENESIS
  - SciFlo similar to JDAF
  - Doppler planner could be used to generate SciFlow “plans”
- IDACT
  - Automatic data type transformations
  - Restricted form of planning for dataflow generation
- DISCOVER
  - Transparent data access
  - Use of ESML
- GeoBrain
  - A geo-tree is a dataflow plan
  - No support (yet) for automatic generation

Related Work (cont'd)

- MVP, COLLAGE
  - Scientific image processing, human in the loop
  - HTN representation, less need for precise causal representation
  - No causal reasoning about data, metadata generation
- Chimera
  - Data tracking, but no support for causal reasoning
- Amphion, AutoBayes
  - Program synthesis using theorem proving
  - More expressiveness than needed for many DP problems
- Internet Softbot
  - Information gathering and changes to world.
  - Could never handle Unix pipes

Ames Research Center

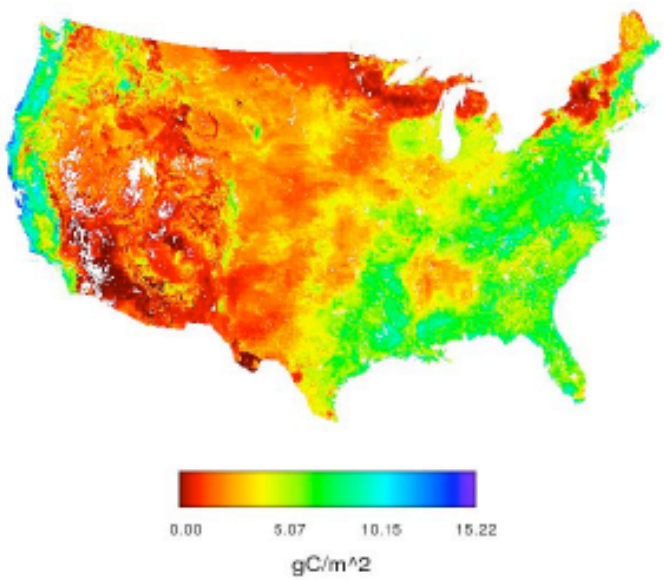
http://ecocast.arc.nasa.gov

ECOLOGICAL FORECASTING

Monitoring, Modeling, and Forecasting the Impacts of Climate Variability and Change on Ecosystems

Daily Ecocast

Daily GPP 6/20/2004



< previous | current >

more images & data

What is Ecocasting?

Ecological forecasting (or 'ecocasting') is the prediction of ecosystem parameters. NASA Ames is developing advanced computing technologies for converting massive streams of satellite remote sensing data into ecocasts that are easy to read and use.

NASA Ames, UWF IHMC, CMU, CSUMB, UMT, UW, and Fetch Technologies are collaborating to develop a distributed computing architecture for the production of ecocasts from satellite remote sensing data and other ancillary data sources. Applications of the Ecocast technology include fire forecasting, crop quality forecasting, snowpack and flood monitoring, and identification of anomalies in the carbon cycle and other biospheric processes.

News

Daily updates of biospheric parameters are now available. See below for a selection of available parameters. Or download data and images here.

Nowcasts & Forecasts

- Meteorology
- Hydrology
- Carbon Cycle